

Practica 2.

Lisardo Gayán Tremps

José Luis Melo

3 June, 2019

Contents

1 - Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?	1
2 - Integración y selección de los datos de interés a analizar	2
3 - Limpieza de datos	5
3.1. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos? .	5
3.2. Identificación y tratamiento de valores extremos.	10
4. Análisis de los datos.	11
4.1. Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar)	11
4.2. Comprobación de la normalidad y homogeneidad de la varianza.	11
4.3. Aplicación de pruebas estadísticas para comprar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.	11
5. Representación de los resultados a apartir de tablas y gráficas.	11
6. Resolución del problema. A partir de los resultados obtenidos. ¿cuáles son las conclusiones?. ¿Los resultados permiten responder al problemas?	11
7. Código. Hay que adjuntar el código, preferiblemente en R, con el que se ha realizado la limpieza, análisis y represntación de los datos.	11

1 - Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?

El dataset de Titanic: Machine Learning from Disaster se registran los datos de los pasajeros del famoso trasatlántico y se utiliza para predecir los supervivientes. Los datos estan divididos en dos dataset, uno de test y otro entrenamiento, para la creación de modelos de predicción.

2 - Integración y selección de los datos de interés a analizar

Se importan los datos. Primero el dataset train.

```
datostrain <- read.csv("./data/train.csv", stringsAsFactors = FALSE, na.strings = c("NA", ""))
str(datostrain)
```

```
## 'data.frame':    891 obs. of  12 variables:
## $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
## $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
## $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
## $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
## $ Sex        : chr  "male" "female" "female" "female" ...
## $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
## $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
## $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
## $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
## $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
## $ Cabin      : chr  NA "C85" NA "C123" ...
## $ Embarked   : chr  "S" "C" "S" "S" ...
```

Se observa como consta de 891 muestras y 12 variables, entre ellas Survived.

Posteriormente el dataset test.

```
datostest <- read.csv("./data/test.csv", stringsAsFactors = FALSE, na.strings = c("NA", ""))
str(datostest)
```

```
## 'data.frame':    418 obs. of  11 variables:
## $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
## $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
## $ Name       : chr  "Kelly, Mr. James" "Wilkes, Mrs. James (Ellen Needs)" "Myles, Mr. Thomas Francis"
## $ Sex        : chr  "male" "female" "male" "male" ...
## $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
## $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
## $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
## $ Ticket     : chr  "330911" "363272" "240276" "315154" ...
## $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
## $ Cabin      : chr  NA NA NA NA ...
## $ Embarked   : chr  "Q" "S" "Q" "S" ...
```

Se observa como tiene 418 muestra, y 11 variables. La variable Survived no aparece porque es la que se tiene que predecir.

A continuación, a la hora de fusionar los datos caben dos posibilidades, asignar “NA” a la variable datostest\$Survived o no considerar los datos de survived en train. Se importan, fusionan los datos y se revisa la estructura inicial de los datos.

```
datostest$Survived <- NA
datos <- rbind(datostrain, datostest)
str(datos)
```

```
## 'data.frame': 1309 obs. of 12 variables:
## $ PassengerId: int 1 2 3 4 5 6 7 8 9 10 ...
## $ Survived : int 0 1 1 1 0 0 0 0 1 1 ...
## $ Pclass : int 3 1 3 1 3 3 1 3 3 2 ...
## $ Name : chr "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
## $ Sex : chr "male" "female" "female" "female" ...
## $ Age : num 22 38 26 35 35 NA 54 2 27 14 ...
## $ SibSp : int 1 1 0 1 0 0 0 3 0 1 ...
## $ Parch : int 0 0 0 0 0 0 0 1 2 0 ...
## $ Ticket : chr "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
## $ Fare : num 7.25 71.28 7.92 53.1 8.05 ...
## $ Cabin : chr NA "C85" NA "C123" ...
## $ Embarked : chr "S" "C" "S" "S" ...
```

A continuación comprobamos los datos que faltan

```
# Busco primero qué variables tienen valores perdidos
missing_numbers <- sapply(datos, function(x) {sum(is.na(x))})
kable(data.frame(Variables = names(missing_numbers), Datos_faltantes= as.vector(missing_numbers))) %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

Variables	Datos_faltantes
PassengerId	0
Survived	418
Pclass	0
Name	0
Sex	0
Age	263
SibSp	0
Parch	0
Ticket	0
Fare	1
Cabin	1014
Embarked	2

Podemos observar, que en Survived, salen los 418, que tenemos que predecir, por lo que todos los valores de train están informados.

A continuación se detallan las variables y su tipo inicial, este último, se modificara para su mejor analisis.

```
# datostrain1 <- datostrain[,-2]
# data <- rbind(datostrain1, datostest) # Fusion datasets
data <- datos[,-2]
str(data)
```

```
## 'data.frame': 1309 obs. of 11 variables:
## $ PassengerId: int 1 2 3 4 5 6 7 8 9 10 ...
## $ Pclass : int 3 1 3 1 3 3 1 3 3 2 ...
## $ Name : chr "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
## $ Sex : chr "male" "female" "female" "female" ...
## $ Age : num 22 38 26 35 35 NA 54 2 27 14 ...
## $ SibSp : int 1 1 0 1 0 0 0 3 0 1 ...
## $ Parch : int 0 0 0 0 0 0 0 1 2 0 ...
## $ Ticket : chr "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
```

```
## $ Fare      : num  7.25 71.28 7.92 53.1 8.05 ...
## $ Cabin     : chr   NA "C85" NA "C123" ...
## $ Embarked  : chr   "S" "C" "S" "S" ...
```

```
tipos <- sapply(data, class)
kable(data.frame(Variables = names(tipos), Tipo_Variable= as.vector(tipos))) %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

Variables	Tipo_Variable
PassengerId	integer
Pclass	integer
Name	character
Sex	character
Age	numeric
SibSp	integer
Parch	integer
Ticket	character
Fare	numeric
Cabin	character
Embarked	character

Las variables, que no tienen datos faltantes, class y sex, se convertirán a factor. La variable cabin tiene muchos datos faltantes, así que en un primer momento no se utilizará.

```
#data$Age <- as.integer(data$Age)
data$Pclass <- as.factor(data$Pclass)
data$Sex <- as.factor(data$Sex)
#data$Embarked <- as.factor(data$Embarked)
#data$Cabin <- as.factor(data$Cabin)

tipos_new <- sapply(data, class)
kable(data.frame(Variables = names(tipos_new), Tipo_Variable= as.vector(tipos_new))) %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

Variables	Tipo_Variable
PassengerId	integer
Pclass	factor
Name	character
Sex	factor
Age	numeric
SibSp	integer
Parch	integer
Ticket	character
Fare	numeric
Cabin	character
Embarked	character

Una vez modificadas los tipos de valores se resume que:

```
summary(data)
```

```
## PassengerId  Pclass      Name      Sex      Age
## Min.      :    1    1:323  Length:1309  female:466  Min.      : 0.17
```

```
## 1st Qu.: 328    2:277    Class :character    male :843    1st Qu.:21.00
## Median : 655    3:709    Mode  :character                Median :28.00
## Mean   : 655                                Mean   :29.88
## 3rd Qu.: 982                                3rd Qu.:39.00
## Max.   :1309                                Max.   :80.00
##                                             NA's   :263
##
##      SibSp      Parch      Ticket      Fare
## Min.   :0.0000    Min.   :0.000    Length:1309    Min.   : 0.000
## 1st Qu.:0.0000    1st Qu.:0.000    Class :character    1st Qu.: 7.896
## Median :0.0000    Median :0.000    Mode  :character    Median : 14.454
## Mean   :0.4989    Mean   :0.385                    Mean   : 33.295
## 3rd Qu.:1.0000    3rd Qu.:0.000                    3rd Qu.: 31.275
## Max.   :8.0000    Max.   :9.000                    Max.   :512.329
##                                             NA's   :1
##
##      Cabin      Embarked
## Length:1309      Length:1309
## Class :character    Class :character
## Mode  :character    Mode  :character
##
##
##
##
```

PassengerId: Variable de tipo entero que contiene el id del pasajero, no existen valores nulos o perdidos.

Pclass: Variable de tipo factor con la categoría asignada al pasajero, no existen valores nulos o perdidos.

Name: Variable de tipo texto con el nombre del pasajero, no existen valores nulos o perdidos.

Sex: Variable de tipo factor con el género del pasajero (masculino, femenino), no existen valores nulos o perdidos.

Age: Variable de tipo numérico que especifica la edad del pasajero, **existen 263 valores nulos**.

SibSp: Variable de tipo entero que especifica el número de hermanos/esposa a bordo, no existen valores nulos o perdidos.

Parch: Variable de tipo entero que especifica el número de padres/hijos a bordo, no existen valores nulos o perdidos.

Ticket: Variable de tipo texto que indica el número de ticket, no existen valores nulos o perdidos.

Fare: Variable de tipo número que especifica la tarifa pagada, **existe 1 valor nulo**.

Cabin: Variable de tipo factor donde se especifica la cabina asignada, **existen 1014 valores perdidos**.

Embarked: Variable de tipo factor que indica el puerto de embarque, **existen 2 valores perdidos**.

3 - Limpieza de datos

3.1. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

Volvemos a mostrar los datos que contienen ceros o elementos vacíos.

```
# Busco primero qué variables tienen valores perdidos
missing_numbers <- sapply(datos, function(x) {sum(is.na(x))})
kable(data.frame(Variables = names(missing_numbers), Datos_faltantes= as.vector(missing_numbers))) %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

Variables	Datos_faltantes
PassengerId	0
Survived	418
Pclass	0
Name	0
Sex	0
Age	263
SibSp	0
Parch	0
Ticket	0
Fare	1
Cabin	1014
Embarked	2

De las variables existentes a continuación se especifican aquellas que contienen valores perdido o nulos.

- Age: *existen 263 valores nulos.*

Para imputar valores de **edad**, hay varias opciones desde la más sencilla que sería asignar la media a otras opciones como la propuesta en <http://jstatsoft.org/article/view/v045i03> Data Analysis with R, A comprehensive guide to manipulating, analyzing and visualizing data in R Pag 373 - 386

Se puede analizar si es mejor imputar por la media, mediana u otro método (rpart o mice)

- Fare: *existe 1 valor nulo.*

Para imputar valores **Fare**

Dado que unicamente hay un valor perdido, es posible imputarlo por la media o la mediana en base al puerto de embarque “S” y la clase “3”

```
data[is.na(data$Fare),]
```

```
##      PassengerId Pclass      Name Sex Age SibSp Parch Ticket
## 1044      1044      3 Storey, Mr. Thomas male 60.5      0      0 3701
##      Fare Cabin Embarked
## 1044   NA   <NA>      S
```

```
M_fare<- subset(data,data$Pclass == '3' & data$Embarked == 'S')
mean(M_fare$Fare, na.rm = T)
```

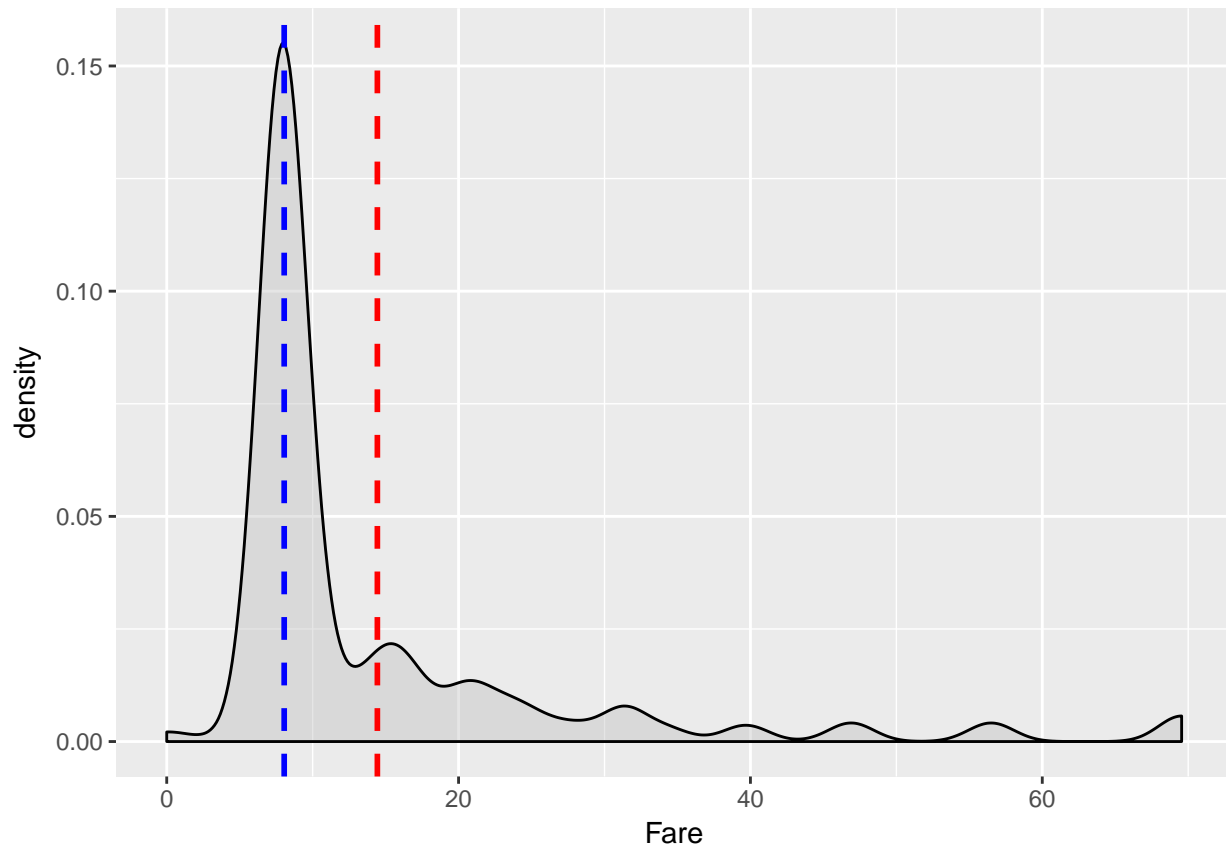
```
## [1] 14.43542
```

```
median(M_fare$Fare, na.rm = T)
```

```
## [1] 8.05
```

```
ggplot(M_fare, aes(x = Fare)) +
  geom_density(fill = 'grey', alpha=0.4) +
  geom_vline(aes(xintercept=median(Fare, na.rm=T)),
    colour='blue', linetype='dashed', lwd=1) +
  geom_vline(aes(xintercept=mean(Fare, na.rm=T)),
    colour='red', linetype='dashed', lwd=1)
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density).
```



Observamos como al realizar la gráfica nos dice que hay un valor nulo.

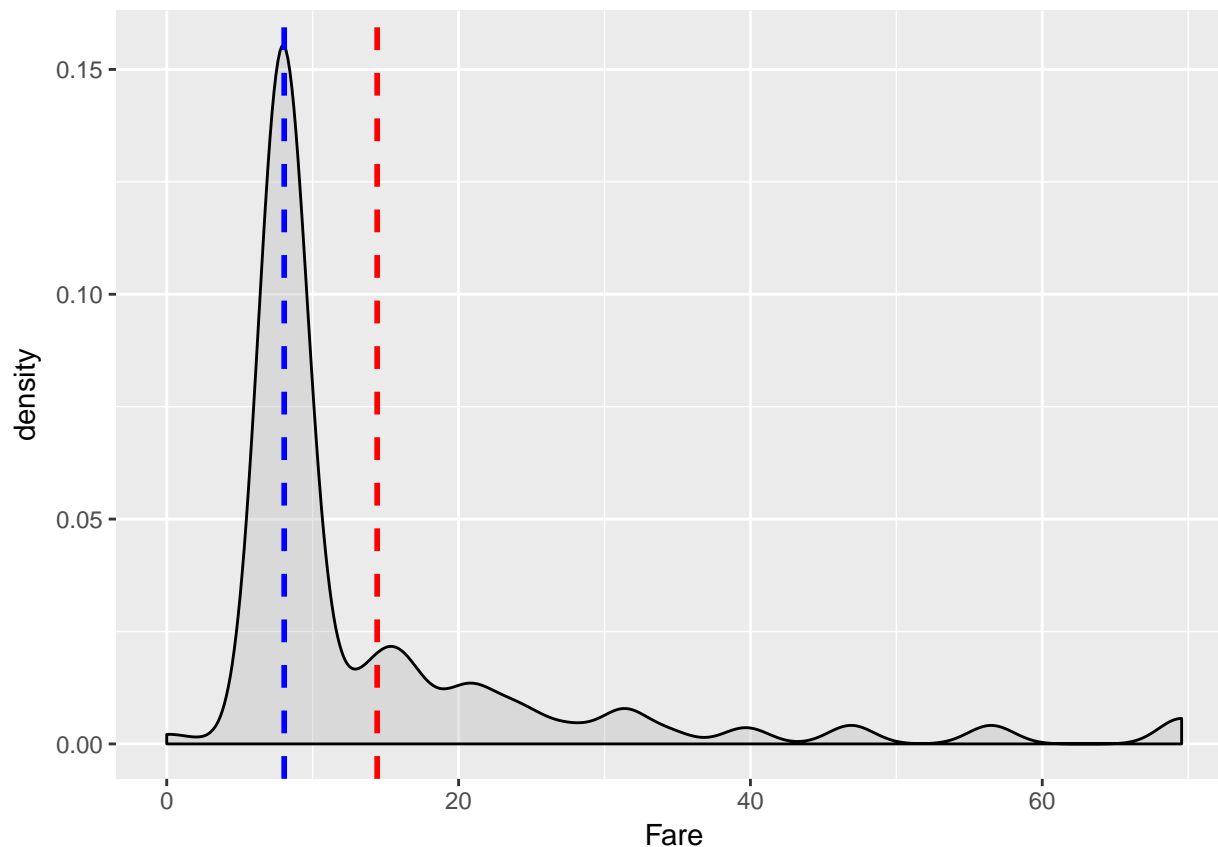
La tarifa de 8.05 coincide con la mediana de los pasajeros de tercera clase que embarcaron en S, por lo que se podría imputar este valor.

```
data$Fare[c(1044)] <- 8.05
data[1044,]
```

```
##      PassengerId Pclass      Name Sex Age SibSp Parch Ticket
## 1044         1044      3 Storey, Mr. Thomas male 60.5    0    0  3701
##      Fare Cabin Embarked
## 1044 8.05  <NA>      S
```

Volviendo a representar

```
M_fare<- subset(data,data$Pclass == '3' & data$Embarked == 'S')
ggplot(M_fare, aes(x = Fare)) +
  geom_density(fill = 'grey', alpha=0.4) +
  geom_vline(aes(xintercept=median(Fare, na.rm=T)),
    colour='blue', linetype='dashed', lwd=1) +
  geom_vline(aes(xintercept=mean(Fare, na.rm=T)),
    colour='red', linetype='dashed', lwd=1)
```



Ahora ya no sale que haya un valor nulo en fare.

- Cabin: *existen 1014 valores perdidos.*

Para imputar valores **Cabin**

Esta variable tiene muchos valores perdidos, se podría conseguir predecir la cubierta asignada al pasajero pero es un dato que poco beneficio podría traer ya que se puede realizar el analisis con la combinación entre la tarifa y la clase del pasajero.

- Embarked: *existen 2 valores perdidos.*

Para imputar valores **Embarked**

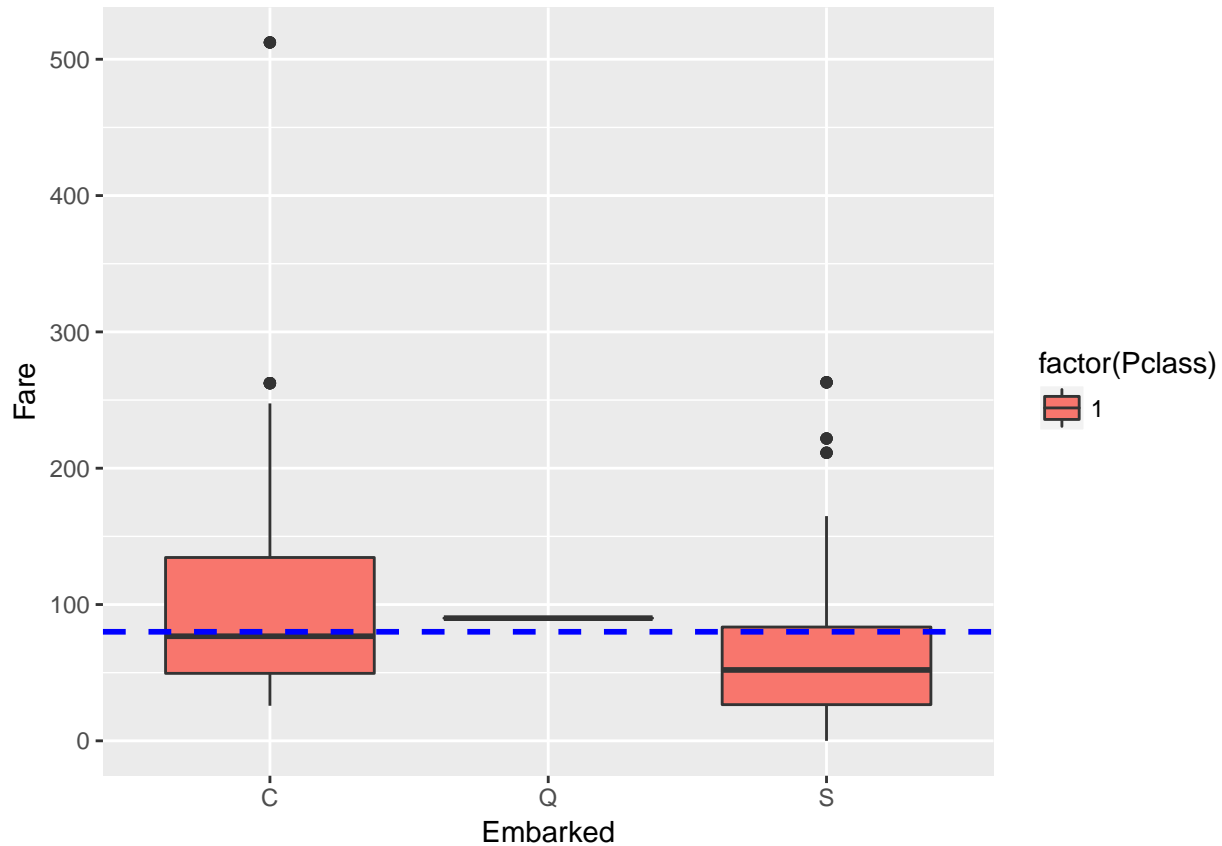
Mostramos los valores perdidos

```
data[is.na(data$Embarked),]
```

```
##      PassengerId Pclass                Name  Sex
## 62             62      1             Icard, Miss. Amelie female
## 830            830      1 Stone, Mrs. George Nelson (Martha Evelyn) female
##      Age SibSp Parch Ticket Fare Cabin Embarked
## 62   38     0      0 113572   80   B28    <NA>
## 830   62     0      0 113572   80   B28    <NA>
```

Al ser unicamente dos valores perdidos, se podría sustituir los valores por la media, en base a otros pasajeros de la misma clase y tarifa (Fare). Los pasajeros han pagado una tarifa de 80 y pertenecian a primera clase.


```
embarked_pass_1 <- data %>%
  filter( PassengerId != 62 & PassengerId != 830 & Pclass == 1)
ggplot(embarked_pass_1, aes(x = Embarked, y = Fare, fill = factor(Pclass))) +
  geom_boxplot() +
  geom_hline(aes(yintercept=80),
    colour='blue', linetype='dashed', lwd=1)
```

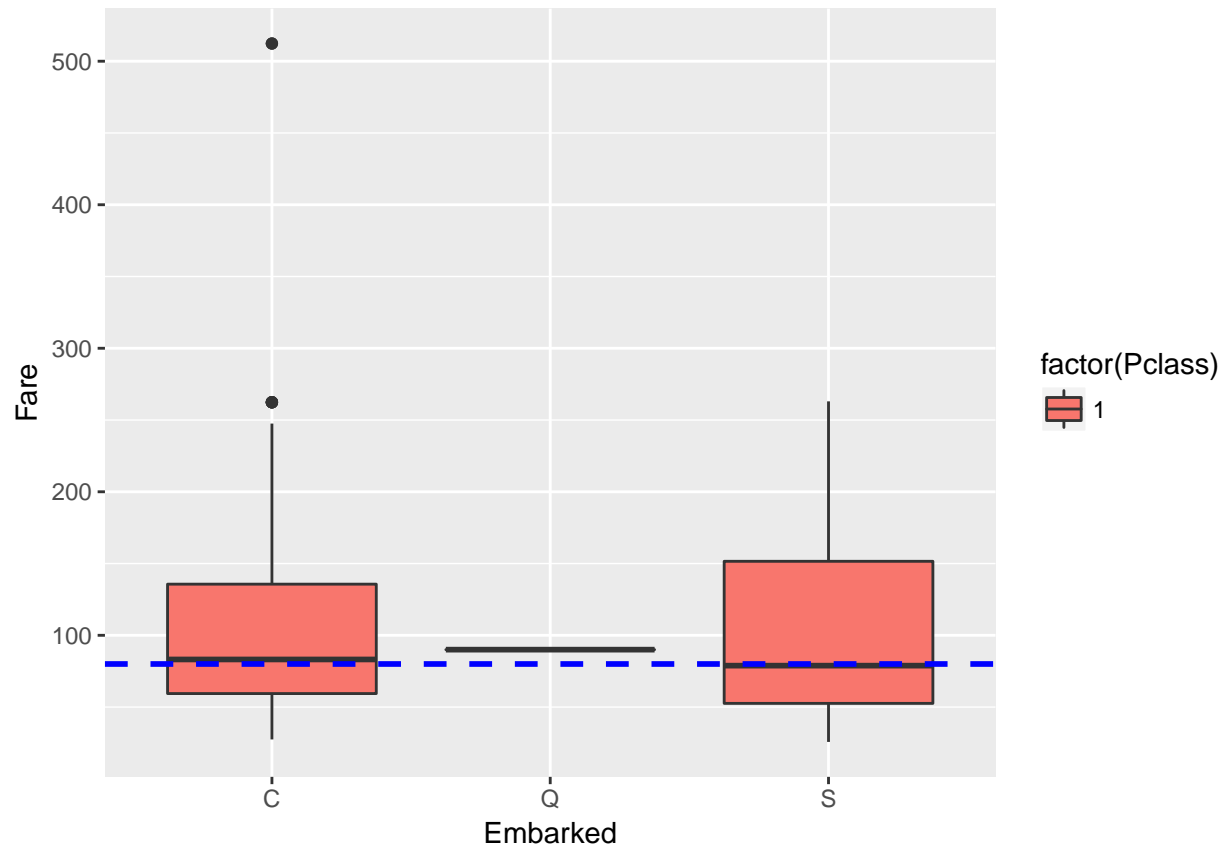


La tarifa de 80 coincide con la media de los pasajeros de primera clase que embarcaron en C, por lo que se podría imputar este puerto.

```
data$Embarked[c(62, 830)] <- 'C'
```

Otra opción, sería considerar también el sexo, ya que principios del siglo XX, no se caracterizaba por una igualdad de hombres y mujeres.

```
embarked_pass_2 <- data %>%
  filter( PassengerId != 62 & PassengerId != 830 & Pclass == 1 & Sex == "female")
ggplot(embarked_pass_2, aes(x = Embarked, y = Fare, fill = factor(Pclass))) +
  geom_boxplot() +
  geom_hline(aes(yintercept=80),
    colour='blue', linetype='dashed', lwd=1)
```



En este caso cualquiera de los 3 puertos tendría una media cercana a 80. Como no creemos que el puerto de embarque este correlacionado con la supervivencia, podemos dejar “C”

3.2. Identificación y tratamiento de valores extremos.

```
##par(mfrow=c(1,3))
##age.bp <- boxplot(data$Age, main= "Edad")
##hist (data$Age, main = "Dist. Edad")
```

4. Análisis de los datos.

4.1. Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar)

4.2. Comprobación de la normalidad y homogeneidad de la varianza.

4.3. Aplicación de pruebas estadísticas para comprar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

5. Representación de los reultados a apartir de tablas y gráficas.

6. Resolución del problema. A partir de los resultados obtenidos. ¿cuáles son las conclusiones?. ¿Los resultados permiten responder al problemas?

7. Código. Hay que adjuntar el código, preferiblemente en R, con el que se ha realizado la limpieza, análisis y represntación de los datos.